

Cloud Computing

Winter Term 2023/2024

Tutorial Session 3



Dominik Scheinert

Distributed and Operating Systems

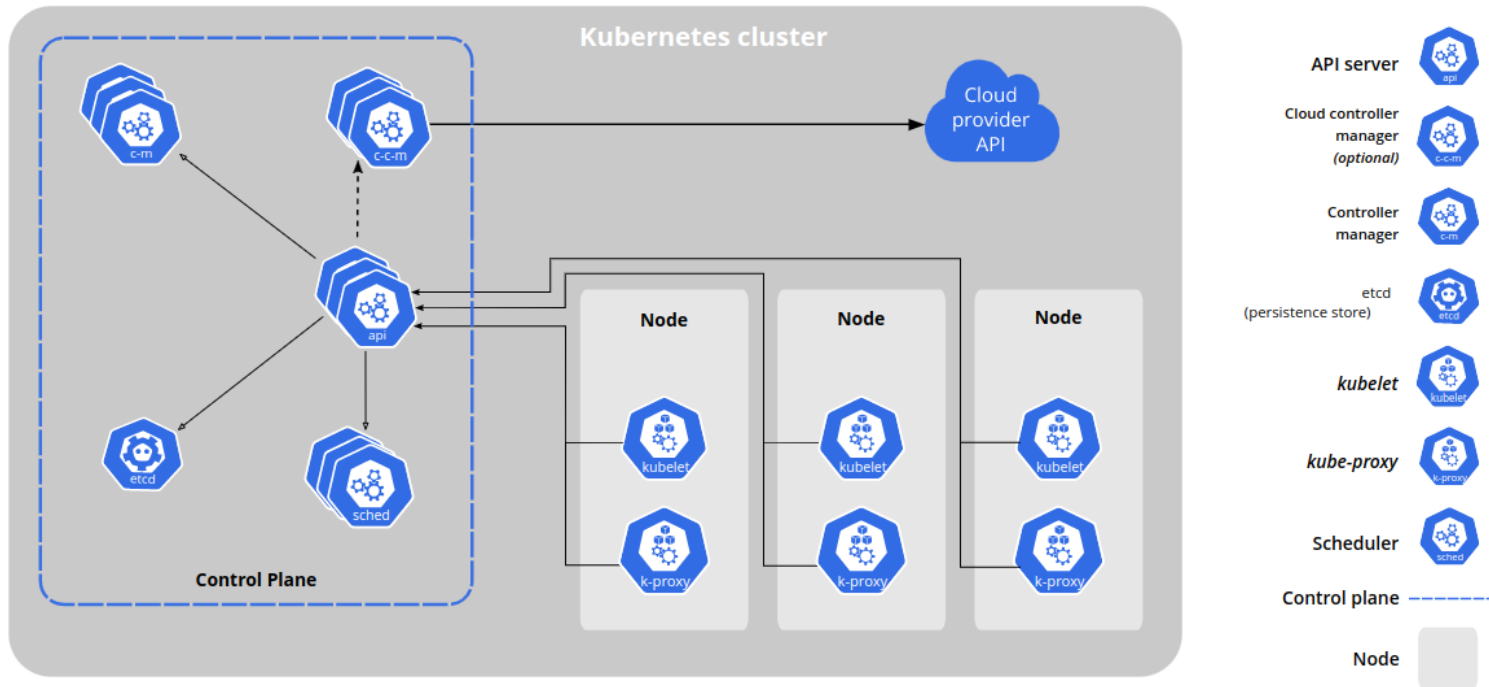
dominik.scheinert@tu-berlin.de

Assignment 3

- Container Orchestration and Infrastructure-as-Code paradigm
- Goal:
 - Deploy two interdependent HTTPS services on a distributed infrastructure
- Tasks:
 1. Set up infrastructure using GCP virtual machines
 2. Install Kubernetes on cluster
 3. Prepare application containers using Docker
 4. Deploy webservices to Kubernetes cluster in an Ansible playbook
 5. Bonus: Implement and demonstrate application autoscaling

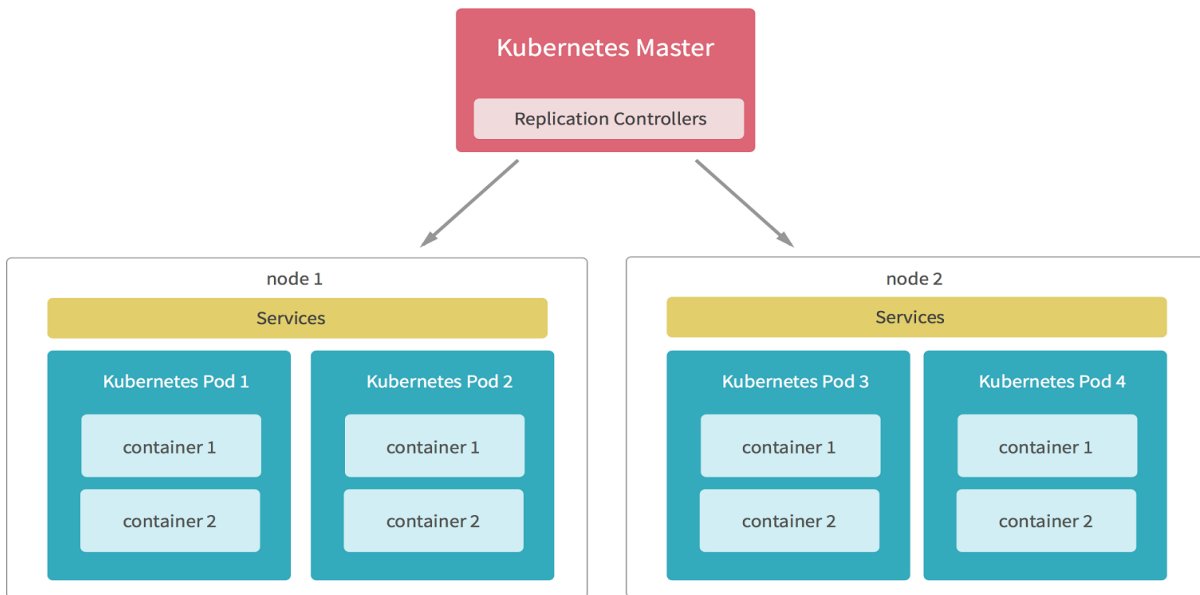
Kubernetes

- Distributed platform for orchestrating containerized applications



Kubernetes

- Distributed platform for orchestrating containerized applications

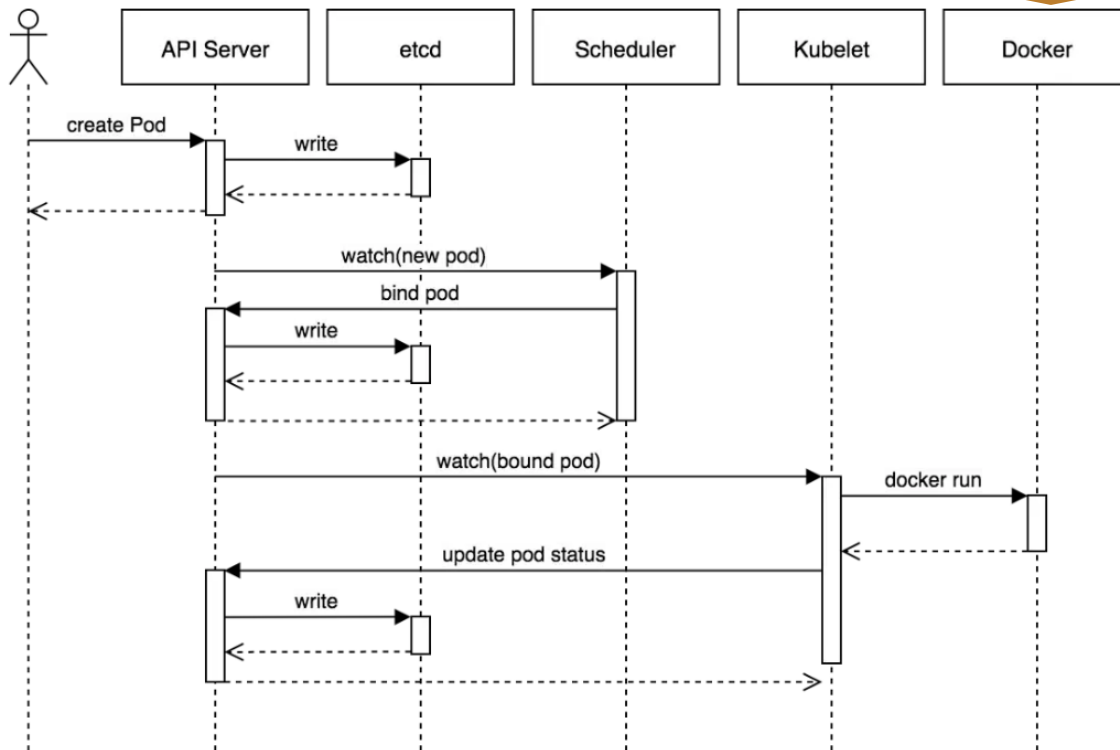


Example Pod Definition

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
    - name: nginx
      image: nginx:1.14.2
      ports:
        - containerPort: 80
```

Kubernetes

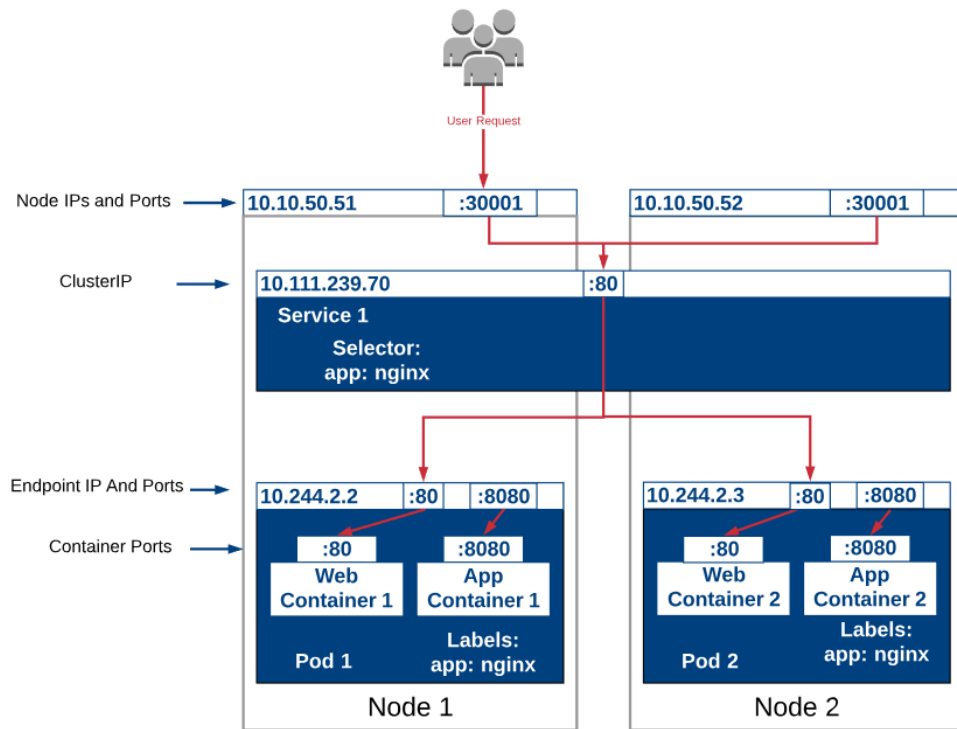
Exemplary container runtime



Kubernetes

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: web-container
          image: nginx
          ports:
            - containerPort: 80
```

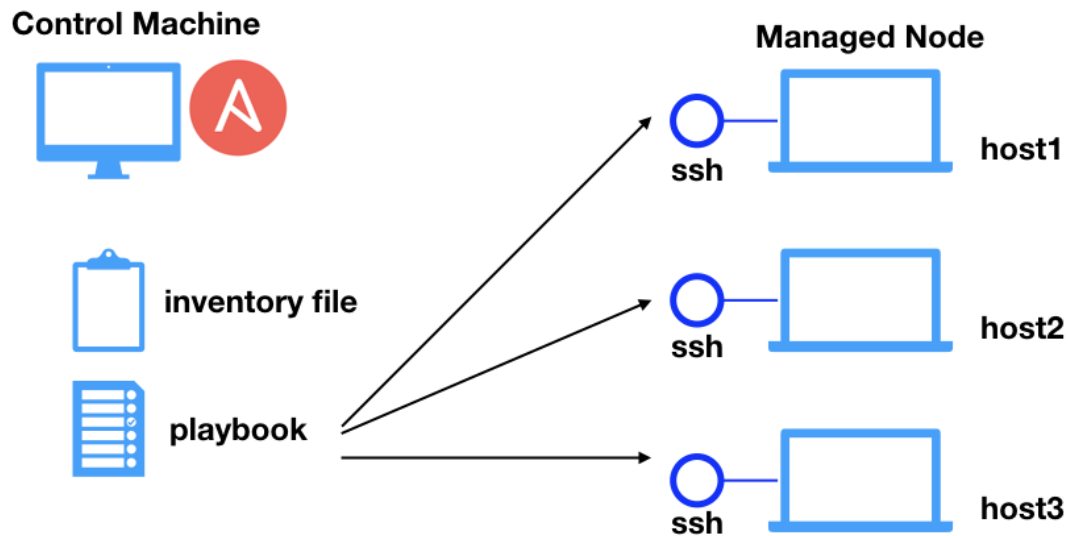
```
apiVersion: v1
kind: Service
metadata:
  name: ingress-nginx
spec:
  type: NodePort
  ports:
    - name: http
      port: 80
      targetPort: 80
      nodePort: 30001
      protocol: TCP
  selector:
    app: nginx
```



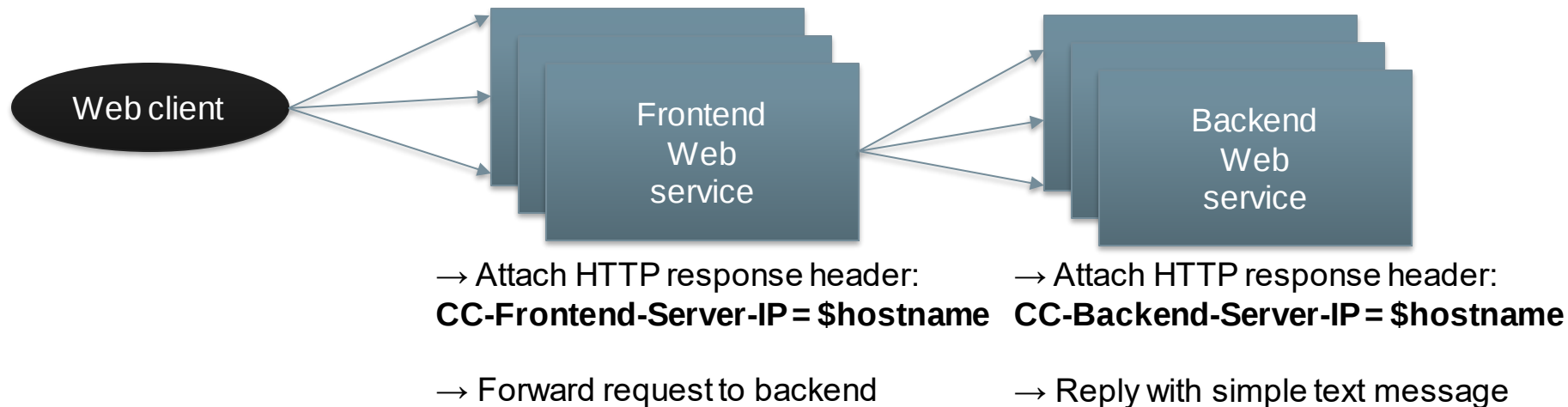
<https://theithollow.com/2019/02/05/kubernetes-service-publishing/>

Ansible

- Declarative description of orchestration tasks on host level
- Idempotent “playbooks”
- Handy modules for all kinds of typical administration tasks



Target deployment



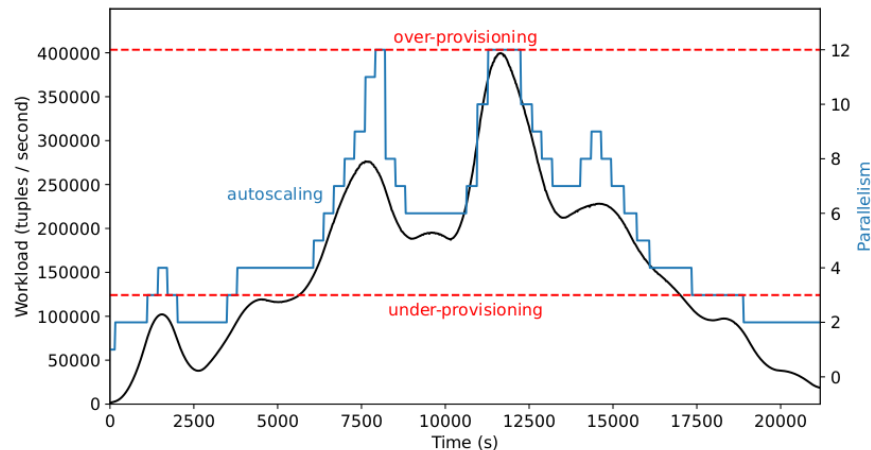
Target deployment



- Send many requests
- Evaluate Response headers

Bonus Task: Application Autoscaling

- Static replication configuration oftentimes
 - inefficient in terms of resources
 - not compliant with defined QoS requirements
- Application autoscaling can be a feasible tool
- In Kubernetes: [Horizontal Pod Autoscaler](#) (HPA)
- Task:
 - Design a use case and scenario
 - Implement a workload generator in order to produce a dynamic workload
 - Configure HPA for the target application
 - Conduct an experiment that demonstrates the autoscaling over time in light of varying demand



Practical Assignment 3

- Due: 14.01.2024
- Summary:
 - Prepare 3 GCP VMs
 - Deploy Kubernetes cluster using Kubespray (Ansible playbook)
 - Prepare simple Docker containers for dummy web service
 - Roll out web service in Kubernetes cluster using an own Ansible playbook
 - Evaluate the deployment with a provided test script
 - Bonus: Implement and demonstrate application autoscaling via HPA